drawSlide - a drag-and-drop tool for Scout window creation and manipulation

Karl King

kgk@dcs.warwick.ac.uk

November 19, 2004

1 Introduction

`drawSlide` is a drag-and-drop tool for Scout window creation and manipulation. It forms an extension of the `drawScout` tool, developed as part of a third year project on timetabling, and supports additional functionality for use as a slideshow presentation tool. Still under development, it can be used for a range of purposes, many of which have yet to be fully explored. Originally developed to overcome the tedium involved in constantly "asking" Eden for current values of observable in model exploration, it can be used to aid model intelligibility, model development, interface development, and even usability of existing models. It can also serve as a simple aid for understanding the Scout notation. It currently supports the creation and direct manipulation of `TEXT`, `IMAGE`, `TEXTBOX`, and `DONALD` Scout windows.

It is often desirable to use `drawSlide` in conjunction with an existing model, or a model under development, and the following two features of the tool help to ensure that such models can be loaded into the same Tkeden session without conflict:

- `drawSlide` operates within its own Scout screens, leaving the standard Scout screen free.
- Everything in the `drawSlide` script has the prefix '`DS_`' (to combat namespace issues)

2 drawSlide Overview

When you first load the `drawSlide` script, you are presented with two additional screens, the `drawingBoard`, and the `modeSwitcher` (see Figure 1). It is relatively easy to use, but this section will provide a brief walkthrough, adding some notes on what happens beneath the interface.

To create a scout window, click on the `Create` button, and select a window type using the adjacent buttons. Click on the `drawingBoard` screen, drag to the desired position, and release to create the window. On this release action, a window definition is executed, and the window is added to the `drawingBoard` screen. The definition executed for a `TEXT` window is shown on the next page.

Figure 1: The modeSwitcher screen

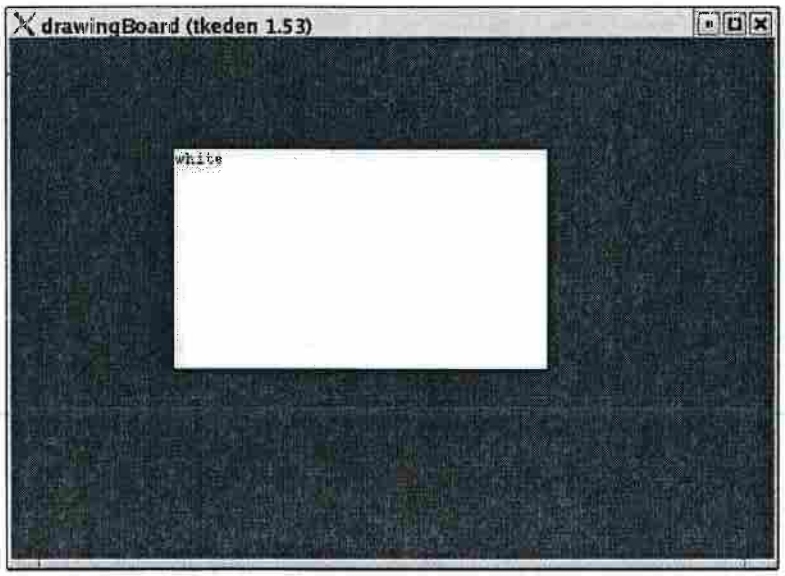

Figure 2: The result of the interaction and redefinition

```
%scout
window DS_win1 = {
    type: TEXT
    frame: ([{DS_win1_X1, DS_win1_Y1}, {DS_win1_X2, DS_win1_Y2}])
    string: DS_win1_string
    font: DS_win1_font
    bgcolor: DS_win1_bgcolour
    fgcolor: DS_win1_fgcolour
    bdcolor: DS_win1_bdcolour
    border: DS_win1_border
    relief: DS_win1_relief
    alignment: LEFT
    sensitive: ON
};
```

The mouse clicks determine the initial values of the point co-ordinate observables in the frame attribute of the window, e.g. `DS_win1_X1` etc. The window definition also includes additional observables for other attributes, such as `DS_win1_bgcolour` and `DS_win1_string`. This is to allow more direct modification of the newly created window. For example, you might want to change the background colour to "white", and to have the current window background colour displayed in the same window. This could be achieved using single definitions, either in Eden, or in Scout:

```
%eden
DS_win1_string is DS_win1_bgcolour;
DS_win1_bgcolour is "white";
```

Contrast this with the way in which entire window definitions have to be re-entered in normal use of Scout. The `drawingBoard` might now look like that presented in Figure 2.

If an `IMAGE` window is created, the `string` attribute is replaced with `pict`. The window is not given a default picture; this requires an additional Scout definition, similar to that below:

```
%scout
DS_win2_pict=ImageFile("gif","myPicture.gif");
```

Having created some windows in `Create` mode, it is then possible to manipulate the windows by selecting `Manipulate` from the `modeSwitcher` screen. By clicking and dragging, the windows can be repositioned on the drawingBoard. Resizing is also possible, but a significant lack of feedback can make the feature somewhat difficult to use initially. It can be achieved by clicking in the very bottom right corner of the window, dragging and releasing. It is important to be aware of certain limitations regarding this resizing functionality though, and these are discussed in section 5. However, it should be sufficient for general use.

3 drawSlide demonstrations

This section is included to give some indication of how `drawSlide` might be used as an aid to the modelling process.

3.1 Basic use

The original tool, `drawScout` was developed to make the observables more readily observable, eradicating the need to continually "ask" Eden for current observable values. By dragging and dropping to create Scout windows, and then introducing definitions that link them to a model loaded underneath, it is possible to provide a convenient display of interesting observables inside the model. The Tkeden history window can be very useful as an aid to remembering window names, as mouseover on sensitive windows causes a redefinition of the mouse position observable, which gets recorded in the history.

3.2 Experimenting with Interface components

One major benefit of `drawSlide` is that it provides a great deal of support for understanding the Scout syntax, and experimenting with different design aspects, such as layout and colour that may assist the development of interface components. `drawSlide` will soon be extended further to support full interface development, but it is relatively difficult to use `drawSlide` for full scale interface development at present, largely due to the lack of appropriate saving functionality.

Buttons can be created by dragging and dropping to create a Scout window, and then introducing a triggered procedure to carry out a series of redefinitions, e.g.

```
%eden
proc redefinitions_to_be_made_on_click : DS_win1_mouse_1 {

    if(DS_win1_mouse_1[2]==5 && DS_mode==DS_useMode)
    {
        definition1;
        definition2;
        ..........
        definitionN;
    }
}
```

It is also possible to create a window and accompanying button for use as a text input window, much like the large input window, as used in previous CS405 lectures. To do this, simply create a `TEXTBOX` and a `TEXT` window and supply Eden with a procedure to execute the contents of the `TEXTBOX` on click actions in the `TEXT` window:

```
%eden
DS_win2_string is "ENTER";

proc execute_Contents_of_DS_win1 : DS_win2_mouse_1 {
```

```
        if(DS_win2_mouse_1[1]==1 && DS_win2_mouse_1[2]==5 && DS_mode==DS_useMode)
        {
           if(DS_win1_TEXT_1#>0)
               execute(DS_win1_TEXT_1);
        }
}
```

Notice the use of `DS_mode==DS_useMode` - this has simply been included to ensure that the contents are not executed when simply manipulating the ENTER button.

3.3 Suggested Use

The major benefit of using the `drawSlide` tool is that it provides a means of exploring the Scout notation, and experimenting with interface designs, and interface component design. Understanding gained from such an exercise can then be applied in building your own interfaces.

4 Tailoring drawSlide

Certain features of drawSlide are easy to change by adding definitions to, or editing definitions in, the drawSlide script. This section is included to provide some information about the things that you might want to change:

- A set of default values of the form `DS_default_attributeName` are included at the top of the drawSlide script. These can be redefined so that the next window created uses these new values.
- To change the size of the drawingBoard, and the drawingBoard screen, copy the drawSlide-1.0 script to your home directory, and edit the screen size observables, `DS_width`, and `DS_height`.
- It may be desirable to lock a created window in a certain position, so that it is not possible to move it in `Manipulate` mode. To do this it is necessary to redefine the entire window, changing the `sensitive` attribute from `ON`, to `OFF`.

5 Limitations of drawSlide

There are a few limitations that you should be aware of if you are intending to use the drawSlide tool.

- One of the major limitations at present is the lack of an appropriate saving function such that . A suitable saving function is under development though, and should be completed within the next few weeks. It is likely that a new version will be released in December 2004. In the meantime, please email kgk@dcs for advice about saving definitions having performed drag-and-drop window creation.
- It may be difficult to keep track of which window is which, due to the similarity of the window names. Use of the Tkeden history window should help to identify those windows that have been created using drag-and-drop, as all have sensitivity turned `ON`. Again, this is something to be addressed in a future version - *the tool was not designed as a drag-and-drop interface creation tool.*
- drawSlide does not currently support xmin, xmax, ymin, ymax attributes of `IMAGE` and `DONALD` windows.

6 An exercise using drawSlide - extending the jugs interface

Yes - it's jugs again! The following exercises should provide a useful introduction to practical use of the drawSlide tool:

6.1 Exercise 1 - Getting Familiar with drawSlide

Load drawSlide by typing the following into a terminal window:

```
tkeden ~wmb/public/projects/tools/drawSlide/drawSlide-1.0
```

1. Create four `TEXT` windows on the drawingBoard, and experiment with repositioning and resizing the windows.
2. Try introducing some definitions for, and dependencies between their attributes in Eden, for example
   ```
   %eden
   DS_win2_string is "bananas"; DS_win1_string is DS_win2_string;
   DS_win1_bgcolour is DS_win2_bgcolour=="red"? "blue":"green";
   ```
3. Introduce a dependency that makes `DS_win3_X1` dependent on `DS_win4_X1`, and try repositioning `DS_win4` in `Manipulate` mode. What happens when you reposition `DS_win3`, and why?
4. Add additional definitions to give window `DS_win4` the same dimensions as `DS_win3`, and to position the windows alongside each other.

6.2 Part 2 - Loading models underneath drawSlide

Having gained some familiarity with drawSlide, it's now time to load the jugs model underneath. Load the Run.e file and the jugs.disp.d file (a Donald interface to the model) into the Tkeden session, from the directory:

```
/dcs/emp/empublic/projects/jugsBeynon1988/
```

1. Click on `New Slide` (on the `modeSwitcher` screen), and create a Donald window.
2. Query Scout for the Donald window definition (the Tkeden history window should help you to identify the window name), copy and paste this into the Tkeden input window, and edit the `pict` attribute line to read:

   ```
   pict: "DoNaLD"
   ```

   (You may need to switch to manipulate mode and click on the window to force the redrawing of the window.)
3. Refer to the tkeden history window whilst clicking on the functional buttons presented on the jugs interface (the main Scout screen). It is possible to create copies of existing buttons with a simple scout definition:

   ```
   DS_win5=windowName;
   ```

4. Create five additional TEXT windows, and introduce definitions to link them to the jugs interface buttons. Move the Donald window to a more appropriate position (if necessary) in `Manipulate` mode.

You should now have your very own interface to the jugs model. Parts 3 and 4 consider possible extensions.

6.3 Part 3 - Creating your own buttons

Create 4 additional Text windows, and introduce triggered procedures that increment and decrement the content and capacities of jugs A and B in response to clicks with the third, and first mouse buttons respectively. Section 3 of this document provides some information on creating triggered procedures.

(HINT: To do this, it may be helpful to refer to the history window whilst performing clicking actions on your newly created windows.)

6.4 Part 4 - The Finale: Creating your own text input window for redefinition

You can now create your own text input window to control the jugs model. Create one `TEXTBOX` window, and a `TEXT` window to act as the submit button. Write a procedure to execute the contents of the `TEXTBOX`, triggered by mouse clicks in the `TEXT` window.

7 Where to find drawSlide

drawSlide is currently available from the following locations:

```
~wmb/public/projects/tools/drawSlide-1.0/
```

```
http://www.dcs.warwick.ac.uk/~kgk
```

If you do intend to use drawSlide, please email me (kgk@dcs) regardless of whether or not you are having difficulty using the tool as your feedback will be most valuable in future development of the tool. In addition, I will keep you up-to-date on the latest developments/answer your queries.